*Science in Context* **12**, 2 (1999), pp. 293–316

MARTINA MERZ

# Multiplex and Unfolding: Computer Simulation in Particle Physics

**The Argument**

What kind of objects are computer programs used for simulation purposes in scientific settings? The current investigation treats a special case. It focuses on "event generators," the program packages that particle physicists construct and use to simulate mechanisms of particle production. The paper is an attempt to bring the multiplex and unfolding character of such knowledge objects to the fore: Multiple meanings and functions are embodied in the object and can be drawn out selectively according to the requirements of a work setting. The object's conceptual complexity governs its application in some contexts, while the object is considered a mere "black box," transparent and ready-to-hand, in others. These two poles span a full spectrum of object aspects, functions, and conceptions. Event generators are ideas turned into software, testing grounds for models, just a tool to study the performance of a detector, etc. The object's multiplex nature is submitted to negotiation among different actors.

> Questions were to be put to the Pythia, the "Priestess" or "Prophetess" of the Oracle. In fact, she was a local woman, usually a young maiden, of no particular religious schooling. Seated on a tripod, she inhaled the obnoxious vapours that seeped up through a crevice in the ground. This brought her to a trance-like state, in which she would scream seemingly random words and sounds. It was the task of the professional priests in Delphi to record those utterings and edit them into the official Oracle prophecies, which often took the form of poems in perfect hexameter. In fact, even these edited replies were often less than easy to interpret. The Pythic oracle acquired a reputation for ambiguous answers.
>
> — Sjöstrand (1995a, 6–7)

## 1. Introduction

What kind of objects are computer programs used for simulation purposes in scientific settings? How should they be conceived of? The current investigation

treats a special case. It focuses on *PYTHIA*, *HERWIG*, *KORALW*, and other "event generators," and on the scientific practice of the physicists who construct, configure, maintain, and run them, as observed at CERN, the European Laboratory for Particle Physics, in Geneva.[1] Event generators are computer program packages, developed and used by particle physicists. The programs "generate events" in the sense that they simulate the mechanisms of particle production that are expected to take place in a collider experiment. This paper is a first attempt to bring the multiplex[2] and unfolding character of such objects to the fore. I will show that event generators are used to tackle different kinds of problems and tasks. They are viewed and conceived of in more ways than one. They are fit into scientific work settings in various ways, and they undergo considerable modification in the course of which their complexity increases.

While research in contemporary physics and other sciences increasingly relies on computer-generated simulations, the significance of such has not been sufficiently analyzed in the science studies literature. One of the exceptions is Peter Galison's study of how the Monte Carlo method emerged in the decades following World War II (Galison 1996; 1997, chap. 8). Most event generators are based on the Monte Carlo method, and Galison's study provides part of their historical background.[3] Galison argues that representatives of various professions (e.g., pure and applied mathematicians, numerical analysts, industrial chemists) had different views about what the Monte Carlo was. He coins the term "trading zone" (as constituted by simulations) to designate the "arena in which radically different activities could be locally, but not globally, coordinated" (Galison 1996, 119; see also Galison 1995, 35ff). Unlike Galison, I am here less concerned with the role of simulation in the coordination of activities that belong to different problem domains (such as, for example, "theory" versus "experiment,"[4] or different disciplines).[5] Instead, I ask whether simulation programs fit certain classifications of scientific objects that cut across the distinction between research fields. In particular, I ask whether event generators may be situated within a spectrum spanned by "epistemic things" and "technological objects," respectively "objects of knowledge" and "instruments." These concept pairings have recently been proposed by Hans-

[1] This study uses ethnographic methods in combination with context analysis of physicists' texts, such as scientific publications, listings of computer programs, program manuals, material presented on home-pages, etc.

[2] "**mul•ti•plex . . . 1.** having many parts or aspects. **2.** manifold; multiple. **3.** of, pertaining to, or using equipment permitting the simultaneous transmission of two or more trains of signals or messages over a single channel." (*Random House Webster's College Dictionary*, 1992).

[3] Monte Carlo programs are named after the technique they are based on: "any technique making use of random numbers to solve a problem" (James 1980, 1147). Called Monte Carlo after the mecca of the casinos — a hint at the random selection of numbers — the method was developed by Metropolis, Ulam, and von Neumann and first applied to problems of nuclear fission (Galison 1996).

[4] The distinction between "theory" and "experiment" as a framework for characterizing computer simulation is used also by Dowling (see her article in this issue) and by Rohrlich (1991). It will remain rather implicit in the present paper.

[5] Cf. Star and Griesemer (1989), who introduce "boundary objects" as an analytic concept to discuss how coherence is developed and maintained across intersecting worlds.

Jörg Rheinberger and by Karin Knorr Cetina in attempts to reach a better understanding of knowledge objects and their dynamics.

## *How to Think of Knowledge Objects*

Hans-Jörg Rheinberger (1992), concerned with experimental reasoning, introduces "epistemic things" and "technological objects" as two different but inseparable components of experimental systems, the smallest functional units of research. Since scientists do not know exactly what they are looking for, the scientific objects or epistemic things whose elucidation is at the center of the research endeavor are yet in the process of "becoming materially defined" (ibid., 310). Examples of epistemic things are chemical reactions and biological functions; Rheinberger speaks of epistemic things also as "question-generating machines" (ibid., 312). In contrast, technological objects, linked to the experimental conditions, are characteristically determined and perform according to known regularities; they are therefore "answering machines" (ibid., 312). The two types of objects are distinguished by their function: they define "places" within the system, and they can change places. They are engaged in a relation of exchange and mutual transformation. For example: "Sufficiently stabilized scientific objects may become transformed into constitutive moments of the experimental arrangement" (ibid., 310f). Rheinberger introduces the distinction between the two object types in order to characterize the "game of innovation" (ibid., 311) — that is, the temporal dynamics which governs experimental practice.

Karin Knorr Cetina (1997) starts from Rheinberger's suggestions in order to get a better grip on objects of expertise. She spells out that these objects should be conceived of neither as instruments nor commodities. Knorr Cetina calls the equation of technological objects with instruments highly problematic "in light of today's technologies, which are simultaneously things-to-be-used and things-in-a-process-of-transformation" (ibid., 10).[6] Computers and computer programs, for example, are "both present (ready-to-be-used) and absent (subject to further research)." Knorr Cetina includes technologies of this kind in the category of epistemic things (or "objects of knowledge" as she also calls them), whereas she excludes "instruments," which she defines as "tools," "available means-to-an-end within a logic of instrumental action." In a sense, she thereby displaces the dividing line that Rheinberger had introduced between the different object types. In consequence, Knorr Cetina characterizes objects of knowledge as "continually unready-to-hand, unavailable and problematic, and also as a possible stage in the career of any thing." Where Rheinberger describes a dynamics of exchange and mutual transformation between the two object types that is effective within an experimental system, Knorr Cetina introduces a different kind of dynamics, that actuated by the

[6] Unless explicitly stated, all further quotes in this paragraph are cited from this same passage.

object's "unfolding ontology": its "lack in completeness of being" and its "capacity to unfold indefinitely" (Knorr Cetina forthcoming, 6).

For a characterization of event generators I will draw on the ideas of both Rheinberger and Knorr Cetina. From Rheinberger I take the idea that an object can occupy different places according to its function and that it can change places. However, I will not confine the discussion to the "smallest integral working units of research" (the experimental systems) as Rheinberger does (Rheinberger 1997, 28). Instead, a characteristic of event generators is that they accomplish different tasks in different contexts — that is, in different work settings and for different actors. Various meanings, functions, viewpoints, are embodied in the object and can be drawn out selectively according to the requirements of a local setting. The object occupies different places and oscillates among them, which gives rise to a third kind of dynamics. In contrast to the one that Rheinberger proposes, this dynamics has no direction in time. Event generators do not turn from one kind of object into another when time passes; they can be at the same time question-generating devices in one setting, and part of "the technical repertoire of the experimental arrangement" in another setting (ibid., 29). Thus a field of tension is defined by two poles: the object's conceptual complexity ("epistemic thing") on the one hand, its black-box features ("technological object") on the other. How can one account for the fact that this characteristic tension seems to be upheld over a considerable period? One may hypothesize that the "unfolding ontology" of knowledge objects that Knorr Cetina emphasizes plays a role. Event generators — like the objects Knorr Cetina is concerned with — "are always in a process of being materially defined, they continually acquire new properties and change the ones they have" (Knorr Cetina forthcoming, 6). I want to emphasize that the poles delimit a full spectrum, a multiplicity of aspects under which the object can be perceived of by the actors, which renders it "multiplex." The remainder of the paper is an attempt to depict and detail this spectrum.

## 2. Event Generators and the Context of Particle Physics

An event generator is a computer program (or a subroutine package) of a given size, typically written in the programming language Fortran.[7] As its name indicates, it "generates events." What is an event in this context? Physicists explain by evoking the imagery of a particle physics experiment: Accelerators produce beams of accelerated particles, for example a beam of electrons and a second beam of positrons, which are brought to collide. In the course of the collision between two initial particles — each coming from one beam — new particles are produced and are subsequently detected by particle detectors. One such collision process is called

[7] The event generators of concern here typically have 10–30 kilo lines of code (Sjöstrand 1992, 228). Attempts are under way to rewrite some important event generators (e.g. *PYTHIA*) in C++.

an event (see figure 1). In the case of the Large Electron Collider (LEP) at CERN, up to fifty particles are created in a single event. The particle number per event can increase to up to several thousand in the case of CERN's projected Large Hadron Collider (LHC).

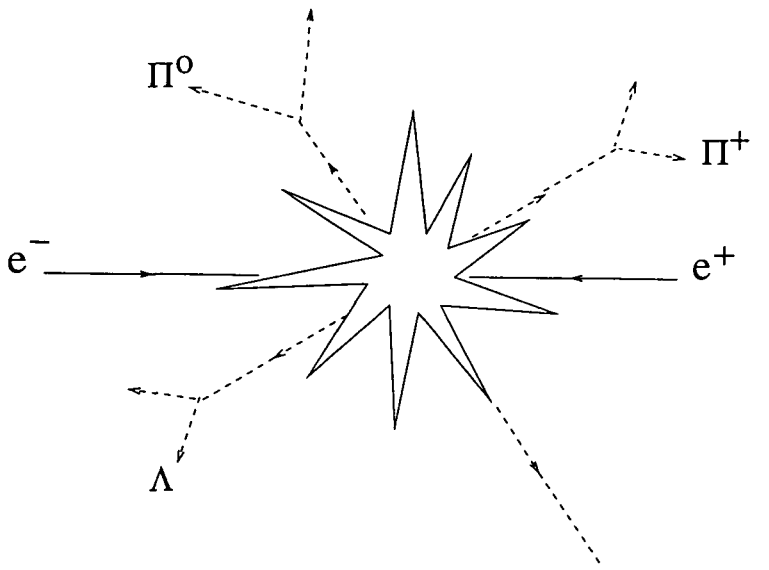


**Figure 1.** An event: In the course of a collision between a beam electron and a beam positron many new particles are created. This process is here only schematically rendered; dashed lines indicate the traces of the produced particles, some of which will decay into further particles.

An event generator "generates events" by simulating the complex mechanisms of particle production that are expected to take place in an experiment. Thus the generator substitutes for the accelerator. The computer program produces "collisions" one by one, as in the "real" experiment. The program generates many events in a single computational session, all are produced with the same initial physical parameters (such as the beam energy). What makes each of them unique is their final output: the listing of all particles created in the event under specification of some of their properties (such as the particle momenta).

Today, a wide range of event generators exists, most of which have been constructed over the last ten to fifteen years.[8] Physicists talk about a "zoology." Generators are distinguished, for example, by their range of applicability, the class of physics processes they simulate, and the "phenomenological models" on which they are based, and it is asked whether they are publicly available or not. In the

[8] Event generators started playing an important role in the late 1970s when three-jet events were searched for and observed in $e^+e^-$ experiments. These observations played an important role in the context of attempts to establish the validity of Quantum Chromodynamics (cf. Pickering 1984, chapter 11).

present paper, I will be more concerned with large "multi-purpose event generators" (also called "general purpose" programs) that are publicly available than with shorter pieces of code for specific purposes.

During their lifetime, event generators undergo considerable modification. In the words of a physicist, the general purpose programs are "unlikely to ever reach the status of a finished product." Their authors regularly prepare new versions and updates. A new version embraces numerous features of the preceding one, resulting in an increasingly complex object that evolves with time. The trajectory of an event generator's development is motivated and influenced by activities within a wider context. For example, the development is often aligned with the evolution, and thus the requirements, of an upcoming experiment. Adapting the code to the new experimental conditions may require that the authors include additional physics processes. Other modifications are based on a better theoretical understanding of the underlying physics. Various kinds of corrections and bug fixes are performed. Because of the ongoing development work, event generators retain their provisional status.

In the world of particle physics, event generators are not isolated entities. Through their embedding relations they can be considered "computing packages" (Kling and Scacchi 1979), a metaphor developed to refer to computer applications as intimately tied to and in conjunction with the social, organizational, and political contexts.[9] Event generators are interfaced with one another, used in parallel, compared, and evaluated. An assessment of the available programs is of great importance for experimentalists who are required to run many different programs. They are framed by other software and incorporate existing code elements as building blocks. They rely on the hardware that provides a physical basis for their activity — such as computers, terminals, and printers — and they are bound to the current development stage of these tools. Event generators further require computing infrastructure — such as networking, program libraries, database management systems, and servers — as well as other services provided, for example, at CERN, by the "Information Technology Division." They rely on the Division's maintenance and development skills, as well as on the skills of the physicists who handle them. They depend on the referential context of both, experiments and theoretical physics, and they require sets of beliefs about what the generators are good for, how they should be written, rewritten, constructed, maintained, tested, and used, and how they can be fit into the continuum of other practices in the world of particle physics.

The current investigation can be seen as an attempt to unpack the package. The attempt will remain partial in that I will focus on certain settings only: those of the

[9] Kling and Scacchi (1979, 108) characterize a "package" as including "not only hardware and software, but also a diverse set of skills, organizational units to supply and maintain computer-based services and data, and sets of beliefs." See Ruhleder 1995 for a discussion of traditional and computer-based artifacts in an academic setting (such as on-line computerized data banks used by classicists) as complex "packages," as characterized by Kling and Scacchi.

physicists who work with event generators.[10] Who are they? Participants sort the scientists who directly handle event generators into "authors" and "users." Authors are the physicists who construct the object and write the code. Users are those who instead run the program to simulate particle production. In contrast to many domains of commercial software development (see, for example, Woolgar 1991), authors and users belong to the same professional community, that of particle physics, although embedded in different work environments (see sections 4 and 5 for a detailed discussion).

### *Authors*

An event generator has only a few authors. Most often, the number does not exceed three. The authors not only construct the object as a piece of programming and develop many of the underlying models, they are also responsible for an important fraction of the maintenance work, since event generators cannot be maintained by technical personnel alone.[11] Event generators are tightly linked to their authors and remain attached to them throughout their lifetimes. The requirement that the code be continuously modified further strengthens the ties between code and author. In most cases, a stable set of authors remains responsible for the object (or for a class of closely related objects) throughout its lifetime.[12] When dividing up the work between the co-workers and when deciding whom to include in the list of authors, authors follow different strategies.[13]

### *Users*

Event generators are today widely employed in the community of particle physicists. The small number of authors contrasts with the increasing number of physicists (and their collaborations) who use the programs in experiments all over the world. Unlike the authors, most users are not committed to a specific generator

[10] For example, the context constituted by the Information Technology Division at CERN will not be considered further.

[11] Some event generators can be obtained from the CERN program library, CERNlib. When users have a technical problem, they may ask the CERNlib staff for help. In other cases, however, computing expertise is not sufficient to identify the problem source and devise a solution. As all-round experts of event generation, the authors therefore remain the users' irreplaceable advisers.

[12] An event generator is only rarely taken over by a new author. In some cases, the group of original authors is joined by a new member.

[13] For example, the multi-purpose code *HERWIG* has six authors, which includes physicists who have never been involved with the programming work as such but have modeled some of the features that later became implemented in the code. In contrast, the "Lund programs" today have one author each: the person responsible for the programming effort and the physics behind it. The originators of the Lund model who also strongly influenced the programs' development were never listed as authors (Sjöstrand 1994, 88). In other cases, model builders and authors coincide, and the physicsts who develop the code's conceptual foundations are responsible for the implementation too.

but apply several of them, often in conjunction with a detector simulation program. The generators play an important role in various phases of an experiment: in an explorative phase in which the "discovery potential" of a proposed accelerator is tested; in the phase of designing the particle detectors in which detector requirements are studied to optimize the design features; prior to the experiment's start-up to devise analysis strategies (i.e., to develop clever ways to separate the signal from the background);[14] and, once the experiment is running, in the phase of physics analysis that relies on a comparison between "real" and "simulated" data. A more detailed description of how simulation is situated in the different stages of an experiment cannot be presented in the frame of this paper (but see the examples in section 5). Suffice it to say that each stage privileges certain forms of simulation activity, but that the multiplex nature of event generators cannot be unfolded simply along the line of an experiment's evolution. For each phase, dedicated work settings (with their own experts, their own patterns of cooperation, etc.) are required. These are embedded in the general framework of a collaboration.[15] Thus event generators are today considered indispensable instruments of knowledge production in all phases of the experiment. In addition, some theoretical physicists use event generators outside of an experiment's context to investigate models.

Although the authors carry the main responsibility for the object and its evolution, users also contribute to its development. Close contact and cooperation between authors and users serve many purposes — for example, to further the object's reliability (users test the object and provide feedback to the authors); to ensure that the users' needs are taken into account in upcoming development phases of the object; to assist in the object being used in appropriate ways (see section 6).

## 3. Physics, Computing, and the Object's Identity: An Author's Perspective

Event generators can be seen as devices to produce numbers, the simulated data, according to the rules of theoretical models that underlie the computer program. Thus an event generator bridges the realm of the empirical — in that it produces a form of data — and the realm of the conceptual — in that it starts out from theoretical models. Evidence for the relation between the object and the realm of the conceptual ("physics" or "theory") is sought by investigating how event generator authors address event generators, physics, theoretical models, etc., in talks and texts such as generator manuals and physics papers.

When authors refer to a particular event generator by its name (e.g., *PYTHIA*, *HERWIG* or *Ariadne*), the name picks out the event generator as a computer

[14] Experimentalists denote as "background" all those unwanted processes among which they need to uncover their signal. This background needs to be fought and overcome (Knorr Cetina 1999).

[15] A collaboration is responsible for an experiment's preparation and performance. In the case of the LHC, a collaboration may comprise more than 1,500 scientific members.

code. For example, "Ariadne is one of the 'Lund family of Monte Carlo programs' and is not a complete event generator" (Lönnblad 1992, 15), or "*HERWIG* is a Monte Carlo package for simulating Hadron Emission Reactions With Interfering Gluons" ("HERWIG information").[16] In contrast, "physics" is mentioned as a separate entity. For example, in a paper published in *Computer Physics Communications*[17] the authors present a "brief review of the physics underlying *HERWIG*" (described in greater detail in a different paper), followed by "a description of the program itself" (Marchesini et al. 1992, 465). Thus, "physics" is distinguished from "the program itself." Other "relevant theoretical background" (ibid., 466) is published in yet other papers. The program's "key components" have a "theoretical basis." Via the basis the program is linked to the realm of physics. The physics underlying the program is also called its "theoretical background." It consists of physics models that are "embodied" in the computer program. What is a model in this context?

> When I write a program, I have to tell details to the computer that are nitty-gritty and somehow it's not part of my conscious picture of the model itself. . . . But then, of course, there are the key assumptions that you are making, I mean, really the new aspects that are the physics aspects and those in addition of course you have to program. So therefore I have this distinction between the model as my way of thinking and then the really typing out of that model in an explicit generator. (Author C; interview)

According to the author, the model is untouched by the constraints of a programming language. The event generator is not. In the process of program construction, the model becomes "embodied." Many aspects of software control need to be dealt with (e.g., questions of compiler compatibility, definition of variables, protection against overflow). These have no equivalent in the model that belongs to the realm of physics alone. Here a comment on the terminology is called for. Particle physicists seem to avoid the term "computer model," which is often used as a synonym for computer simulation programs in other fields. I suggest that subsuming the programs under the label "computer models" would render it difficult to distinguish analytically between the object (the generator) and its physics background (the underlying models). This separation is needed to apprehend the object's multiplex character. One may of course wonder whether also the underlying models themselves or other models that are not embodied in a computer program exhibit features of multiplexity; but this question is not at stake here.[18]

[16] See home-page *http://hepwwwr.rl.ac.uk/theory/seymour/herwig*, last modified on 15 June 1998.

[17] Papers published in this journal are typically the main references to event generators.

[18] Sismondo (forthcoming) seems to argue along this line. He explicates the "multiple domains of models" by presenting the example of the "equilibrium model of island biogeography," which is not a computer model. He claims that "apparently inconsistent interpretations of the model . . . may be mutually supporting."

Specific models are at the "heart" of a program. For example, the Lund string model remains "at the heart of the *PYTHIA/JETSET* programs" (Sjöstrand 1995a, 3). The conceptual center of an event generator, its heart, is a determining factor for a generator as an individual object with a proper identity.[19] What does the heart consist of? And what defines a generator's identity? A comparison between the two multi-purpose generators, *HERWIG* and *PYTHIA/JETSET*,[20] presented by one of the *HERWIG* authors, serves as an illustration:

> The emphasis of HERWIG and JETSET in particular is slightly different. HERWIG is really intended to be a very precise implementation of perturbative QCD. And then has a very simple model of non-perturbative hadronization, kind of, pushed onto the end of it. Whereas in the Lund programs you really came the other way round. They were first thinking about the non-perturbative hadronization and confinement.... And then later they started to think about kind of building perturbative effects into them. (Author A; interview)

*HERWIG* is characterized by the focus of the modeling effort: a very precise model of one part of the event (the "parton shower") is combined with a "simple model" of other parts of the event (the "hadronization"). Thus the emphasis — the fact that one part of the event is neglected in favor of another — distinguishes *HERWIG* from *PYTHIA/JETSET*. Not only are the models different; the principle of organization underlying the code construction — that is, the choice of the building blocks and the way the code is patched together by them — varies among the generators. In the case at hand, the emphasis also lies in different computational schemes. The choice of emphasis — the special features of the code — is strategic, it determines the generator's identity. The two generators complement each other due to the different principles of construction.

The identity that authors and users attribute to an event generator is primarily determined by its conceptual components. It is linked to the principle of construction on which the event generator is based. The principle of construction prescribes, for example, which steps of the event are to be modeled with a high precision. The principle of construction typically relates to the physics choices made while modeling, and not to questions of software management and control. Furthermore, caring for and maintaining an object's identity not only renders it identifiable among other objects; it also introduces a relative constancy over time. Throughout

[19] Several factors contribute to the event generator's individualization: A first factor is the lasting attachment of a small group of authors who shape and configure a specific object. They continuously care for it, treat its problems, cure its weaknesses, advocate and strengthen its potential. The object becomes further individualized by the proper name it carries (see also section 6) and by the specific potential and features through which it becomes further distinguishable from sister objects.

[20] *PYTHIA* and *JETSET* are the two main components of the "Lund Monte Carlo program suite" (Sjöstrand 1994, 74). They were conceived of separately but had thereafter so often been used together and were "gradually coalescing," that in the most recent version they have been merged into one, under the *PYTHIA* label.

the evolution of the code, authors typically stick to "the original intention of the program" and do not revise the principle of construction. The object remains "itself" and keeps its name (although often labeled by a new version number) in the rare cases that its authors change (e.g., work might be handed over to a younger person), after a complete rewrite (i.e., a reorganization of the code) or after other adaptations, updates, and changes of the code.

One may now speculate on the location of "the conceptual" in the event generator by considering the terms "background," "basis," and "heart" at face value. They hint at the following: The "heart" makes things run, the "basis" is essential, and the "background" is the behind-the-scenes work that allows the foreground to exist. The conceptual supports the program in a fundamental yet discreet way (it remains at the back). The authors, then, mediate between backstage and stage. The stage, the foreground (the program) is what users can directly access. Thus the metaphors exemplify another realm of the authors' intellectual contribution. The metaphors also suggest that there is a distinction between those that interfere with the object's most inner parts (the authors) and those who do not (the users). This maps the object conceptions of the different actors: While the necessity to deal with the object's conceptual core renders it an epistemic object to the authors, the inner logic may remain a black box to some users who conceive of the object mainly as a tool (see sections 4 and 5). This is not to say that users in general do not care for the object's conceptual background, as will be detailed below.

## 4. Authors: The Object as Implementation of a Model

The fact that event generators have a theoretical background requires that the authors "implement" the underlying theoretical models in the computer code. The authors need to accomplish the activity of "turning ideas into software," as they say. They are involved in the "practical implementation of the theoretical results" (Bardin and Kleiss 1996, 6).

### *Constructing the Object and Understanding the Model*

Most event generator authors are theoretical physicists, and a considerable fraction of the modeling effort represents their original research. In contrast to what one might expect, implementing theoretical models is not a straightforward activity requiring mainly the skills of a software engineer.

> You have some kind of cycle that you developed something and it's a first approximation but it's not good enough. So when you start to compare with whatever data is available, or when you start to compare with your prejudice about what should be coming out, then you realize that this is not what you

> would have expected. And then, of course, you have to go back and try to understand what were the assumptions that were critical for this feature coming out the way it did. And do I really believe those assumptions that went in there, and how could I check it? (Author C; interview)

Physics and computing efforts are tightly linked and intertwined: work on the conceptual background of the code is closely related to the program development. Whether the first approximation of a model feature is "good enough" will be judged mainly by considering the program's output (once the feature has been turned into software) with respect to expectations and other data. This will then be checked by adapting model and program.

Authors view an event generator first of all as the result obtained when implementing physics models in a computer code. When talking to theoretical physicists, authors sometimes simply refer to their event generator as "the model." Thus the object represents the theoretical concepts on which it is based. The authors are concerned with understanding the model that underlies the generator. Understanding the model requires running the code.[21] By analyzing the simulated data, authors learn more about the model: they learn how it manifests itself in the object. Their focus is on the relation between the event generator and the model(s) (figure 2). In this respect, the authors conceive of the generator mainly as an epistemic object.

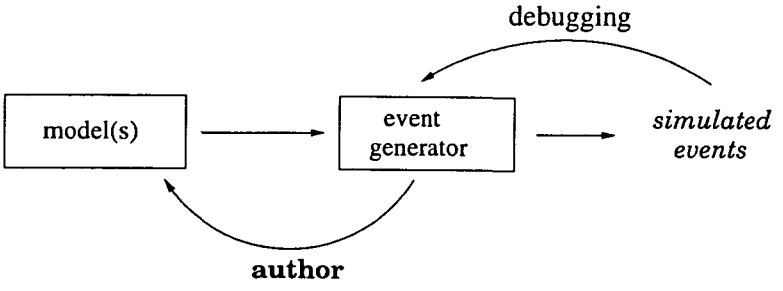


**Figure 2.** Event generator as implementation of a physics model.

### *Preparing a Publicly Available Tool*

Authors have a double agenda: They construct, modify, update, and run the code to suit their own interests (see also below). At the same time they serve the community of users by configuring the generators as suitable instruments. Turning software into "tools available to the public" requires special activities. For example,

[21] Running the program, of course, also serves the purpose of "debugging," a first and important activity after the program has been constructed (see figure 2).

a detailed documentation of the program needs to be written. The program also needs to be constructed in such a way that it will be easy to use and difficult to "misuse." By preparing the object as a publicly available tool, the authors make it possible for the object to be perceived by the users under a multiplicity of aspects that can be selectively drawn out. In the authors' activities, however, the epistemic and the technological aspects of the object remain tightly coupled and interlinked. Authors are responsible for the ongoing development of the event generator as an epistemic object (i.e., concerning the underlying physics) as well as for the work required to maintain it as a software tool (i.e., concerning its technical features).[22] Due to their double agenda, authors are reluctant, and cannot afford, to become too heavily involved in providing service to experimentalists and their collaborations. Authors also struggle against the tendency of users to consider a generator exclusively as a technological object, as if it were a mere tool that could be used in an unproblematic way (see section 6).

### *Doing Physics*

Authors construct the generators not only to serve users by providing a publicly available tool. Most (if not all) authors also use their own code to "do physics"; they therefore develop it as a tool that matches their own interests, adapting it to their specific needs. One of *HERWIG*'s authors explains:

> On the whole we [the authors] are all just working on our own physics areas. Essentially all of us are actually users of HERWIG as well. We don't just provide it as a tool to other people. And so quite a lot of the changes we make are actually associated with things that we are working on at the time. That's really not coordinated at all. It's just as things come up you modify them. (Author A; interview)

Authors point out that the introduction of major new pieces into an event generator is typically motivated by a physics project they have in mind. Doing physics, then, requires, for example, running the program in order to compare the new (underlying) model to others and to analyze the simulated data (see section 5 and figure 3.1). These motivations have a residue in the informality that characterizes the cooperative work of event generator construction among the authors in some cases, as the quote illustrates. Authors claim space for being inventive in their theoretical physics research. They like to be as unconfined as other theoretical physicists are in defining their research topics and projects (Merz 1998).

When explaining the priorities of the theorists among the authors another factor is important. The authors' career paths are aligned with those of their colleagues in

[22] When problems occur, users establish contact with the authors either directly or through the mediation of the contact persons of an experimental collaboration.

theoretical physics. "Monte Carlo work," however, stands apart, in that it is considered by some particle theorists to be work that does not belong to the realm of theoretical physics. Authors believe that "there exists this kind of picture for everybody that the more abstract you are, the more smart you must be," that "you are a good guy" when you have no contact with experimentalists and when you are not involved in anything that could be "applicable in any possible future" (author C). Young authors feel they need to prove that they are capable of contributing to the advancement of theoretical physics; they need to show that they can do "more" than construct Monte Carlo programs and assume responsibility for the maintenance of a software package that is of particular interest to experimentalists and far less so to theorists.

## 5. Users: The Object as Testing Ground, Concretization, Tool, etc.

Users run event generators in order to produce simulated data. Event generators are software tools and the users' main concerns are that these tools be available and reliable. Describing how the activity of event generator construction belonged to neither the world of theory nor to that of experiment, an author put it in the following words:

> Who cares about these Monte Carlo programs? They have to be there. It's like having electricity at CERN. In a sense. (Author B; interview)

The comparison with electricity at CERN suggests that event generators are conceived of by the users as equally ready-to-hand and transparent. It suggests further that an event generator "becomes problematic only when it is unavailable, when it malfunctions, or when it temporarily breaks down" (Knorr Cetina 1997, 10). With these words Knorr Cetina characterizes "equipment" (using Heidegger's term for instruments and following his ideas) in contrast to knowledge objects. I want to argue that in some contexts of application event generators are indeed considered by the actors as "available means-to-an-end" (Knorr Cetina) while in others they exhibit the rich epistemic texture characteristic of knowledge objects. They need not be exclusively one or the other. The object's multiplex character makes it possible for different registers to be pulled in different contexts of application. Particle physicists use event generators for a variety of purposes inside and outside the contexts defined by the various phases of an experiment (mentioned in section 2).

In what follows, I will sketch several examples. The first is devoted to the application of event generators by theoretical physicists; others illustrate how experimental physicists conceive of the object.

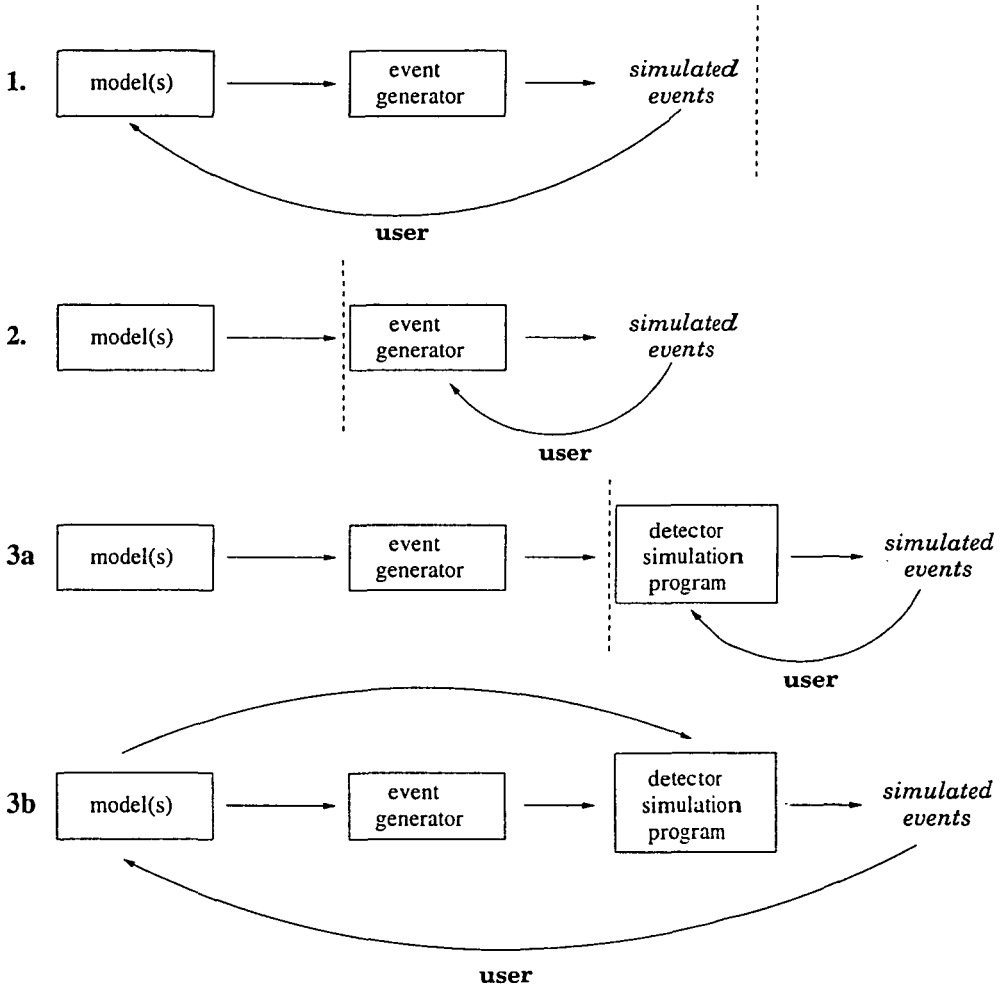


**Figure 3.** Event generator as (1) testing ground for models; (2) concretization of a physics model; (3a) and (3b) tool to optimize the detector design. The vertical line demarcates the actor's field of interest and responsibility from the region that is of lesser importance — for example, because it is taken for granted. The actor's main field of activity is delimited by the arcs. For further explanations see the main text.

## *Testing Ground for a Model*

Event generators are used for investigating different kinds of models. This is important, for example, when physicists explore the potential of a new experiment (e.g., when they ask what kind of "physics" will be accessible with an accelerator of higher energy). Such studies are performed mainly by theoretical physicists and, in particular, by the authors themselves. For instance, physicists use event generators

to study "some new physics." Typically this requires that only part of the code's theoretical background become modified or that a new feature be added.

> The most convenient way to look at the new physics is add the new physics to the Monte Carlo like ours and look what will be the difference in distributions. (Author B; interview)

Physicists might also introduce what are humorously called "crackpot models," "just to check whether one should expect any testable consequences" (Sjöstrand 1995b, 2). Crackpot models are not believed to provide an adequate description of the system studied. In the case at hand, they are introduced rather to investigate whether specific characteristics of a model need to be taken into account. Physicists will ignore features that have no discernible effect or that lead to results which seem to them implausible.

In these examples, the event generator is used as a testing ground for models. One could also say that the event generator is used as a processor that turns model features into numbers, the simulated events (figure 3.1). More than an object, the event generator here represents an environment (or a condition) of investigation. It is in this sense that the following statements of authors should be understood: "Playing a lot with these tools gives you a very good feeling, a very good judgment of physics itself" (author D); "I was in the position of understanding physics better because somehow while doing tests and developing Monte Carlos I had some distributions at the tip of my fingers" (author B). In this situation of application, the event generator is neither just an epistemic nor just a technological object. However, one may say that it constitutes a technological condition incorporating epistemic features: although the model under consideration seems detached from the testing ground, it is also part of and tied to it.

### *Concretization of a Model*

In certain contexts of application, users in experimental physics conceive of an event generator as a "concretization" of the underlying physics models. The users draw a distinction between an abstract model based in the realm of theory and something concrete, something with which concrete things can be done or which produces concrete outcomes that can be further processed. Simulated data look like "real" data — that is, the data types are indistinguishable in their form. As a result, any "experimental question" can be asked and answered with the help of event generators. This interpretation implies that the users can get a grip on the underlying model by dealing with its concrete realization, a situation that is important in the context of data analysis. Here the object's instrumental aspects (its technological features) are emphasized, whereas implementation of the models in the generator is taken for granted and remains unquestioned. The object is used as an instrument that serves an epistemic goal. Figure 3.2 illustrates that analyzing

the simulated events allows users to reach a better understanding of the event generator. For example, an unexpected result can be traced back to a specific piece of the generator by "dissecting" or "opening up" the program. As a concretization of the model, the generator replaces the model: the generator becomes the interface between "physics" (as "theory") and the user.

Authors play on the fact that users conceive of the event generator as a concretization of the underlying models. This is of importance, for example, when authors want to "sell a given model" to the users. Since performing an experimental analysis is very time-consuming, experimentalists need to be motivated to engage in a certain strategy of analysis. A theorist tries to sell a model by showing what, "for example, the model that he is proposing for some new physics would look like if experiments were looking for it" (author D). Sjöstrand's comment that the event generator is "a vehicle for the dissemination of interesting theoretical ideas to the experimental community" (Sjöstrand 1995b, 2) also accentuates that the object transports abstract ideas into the world of concrete data.

### *Just a Tool*

For other users in experimental physics the object is nothing but a tool to study the performance of a future detector and to optimize the detector design: the generator serves as a source of events that are subsequently fed into a detector simulation program. On the one hand, users carefully select a generator according to its suitability for the task at hand and according to the expected physical and software reliability. On the other hand, the object — as a tool — seems transparent to the users since its internal structure and composition are of no particular interest once the selection has been made. In this context of application, the object seems stripped of its conceptual origin (figure 3.3a). The users are merely concerned with the output of the detector simulation. They investigate the simulated events so as to draw conclusions about the features included in the detector model. Eventually, they want to learn how the "real" detector should be designed to ensure that it will be sensitive to the events of interest. In this situation of application, the users seek the event generator as a technological object.

Figure 3.3a needs to be complemented. It is somewhat misleading in suggesting that physics considerations play no role in the phase of detector design. The task of detector construction is more complex than explained above: (a) a detector is composed of several subdetectors, (b) various "physics channels" are to be studied by the same experiment (and, thus, detector).[23] In the phase of detector design, a compromise needs to be sought between the different optimization tasks that

[23] What are physics channels? The simulations of a collision are defined by the type of interacting particles that characterizes an experiment (for example, $e^+e^-$ in an LEP experiment) and by the first step in the reaction (for example $e^+e^- \rightarrow W^+W^-$). These two characteristics define different physics channels in a particle physics experiment.

concern the different subdetectors, the way in which they become merged to a full detector, and the different physics channels. For example, the space assigned to a specific subdetector will be limited due to the space needed for the others. The criterion for convergence of the design specifications is thus not determined solely by the lessons learned when confronting the simulated events with the detector simulation program (as illustrated in 3.3a). Instead, several of these processes overlap and are interlinked in complex ways. This requires that the simulated events be repeatedly measured against the theoretical considerations and physics priorities, which then induce a modification of the detector simulation program (as presented in figure 3.3b). The event generator, however, will remain for the users a technological object.

The examples above illustrate how event generators can be fit into a variety of application contexts in various ways. It is important to note that the order in which the examples were presented does not correspond to the time evolution of a high-energy physics experiment. If one wishes to align the above-described functions with the stages of an experiment's development, one finds that the phase of detector design in which the object is "just a tool" is framed by phases in which the generator's epistemic features are of more direct relevance. For example, the generator will be questioned with respect to the physics on which it is based once the detector design has been fixed, the detector has been constructed, and the first real data are taken:

> The day that you have fixed the [detector] design and you construct, and you start to have real data, at that moment ... you start an intensive test of the generator with reference channels. (User E; interview)[24]

The question suggested above, whether event generators are nothing but transparent instruments in the eyes of the users, can now be answered. Event generators are tools by definition (as described in this section's first lines), but that does not render them transparent in all the situations in which users apply them.

## 6. Cautionary Tales, Recommendations, and Warnings

> It is far too easy to treat it as a black box and I am certainly doing whatever I can to fight that tendency. . . . The experimentalists always have so many other things to worry about. So there is always the danger that people don't take the effort to even learn the basics of generators. (Author C; interview)

The multiplex nature of the object may require work of negotiation and coordination among the different actors: Some users' demand for stability and fixation may

[24] Le jour où on a figé le design [du détecteur] et on construit et on commence à avoir des vraies données, à ce moment-là . . . on commence un test intensif du générateur avec des canaux de référence.

need to be balanced against other actors' preference for change and reconfiguration. Of interest here is yet another aspect of the differing attitudes toward the object: the tension that arises between the object's black-box features (producing results requires only little background knowledge) and its considerable conceptual complexity ("successful use" requires a minimum of understanding). Authors have only limited control over the ways in which the generators are employed (e.g., by "hiding" certain features of the code from users). They try to influence users by advocating a certain understanding of the object: they tell cautionary tales and give recommendations and warnings. This discourse can be seen as a way to (re)negotiate the object's character, which is conceived of differently by the different actors. Authors express their concerns through various means. The following examples are selected from written texts (e.g., event generator overviews or manuals).

### *Need for Critical Examination: "It is therefore extremely foolish . . ."*

Authors list various reasons for the need for caution in using event generators: The intrinsic complexity of the object (and the underlying models), the limited understanding of physics, the imperative use of approximations and the fact that "physics ideas" cannot be implemented in a straightforward way. Given these factors it is negligent to use event generators in an uninformed way.[25] Physicists who rely on an event generator are therefore called upon to first exercise "critical judgement as to its merits and deficiencies" (Knowles and Protopopescu 1993, 651). Another recommendation, similarly motivated, is "not to trust blindly the results of any single event generator, but always to have several cross-checks" (Sjöstrand 1992, 227). Users should base their physics studies on at least two complete and independent programs. Furthermore, the problems that the epistemic openness causes may be amplified by the users' inappropriate way of acting: "An event generator cannot be thought of as an all-powerful oracle, able to give intelligent answers to ill-posed questions" (ibid.).

Another strategy of authors in laboring for a certain understanding of the object is to enroll its name.[26] The story of the historical Pythia, as told by *PYTHIA*'s (the event generator's) author in an appendix to the manual, provides an example.[27] Sjöstrand explains how Apollon founded the Pythic Oracle in Delphi, how questions were put to the Pythia, how her words were not easy to interpret, etc. He then concludes: "The history of the PYTHIA program is neither as long nor as dignified as that of its eponym [but] some points of contact exist" (Sjöstrand

[25] Approximations in the context of computer simulation are discussed by Winsberg (see his article in this issue).

[26] "Local strategies for making a name," especially for machines, in high-energy physics are discussed in Traweek 1992, 446ff.

[27] Event generator authors are responsible for the documentation that accompanies the object, such as manuals.

1995a, 8). The object's name and its story become the gimmick for a cautionary tale. The parallel drawn between the object and its namesake serves to induce recommendations as to how the object should (or should not) be used: Be careful in formulating the questions because "any ambiguities will corrupt the reply"; avoid misinterpreting the answers; and be aware of the fact that the code may not be bug-free. Once again, critical judgment and some understanding are asked for.

### *Limited Liability and Documentation: "No guarantees are given . . ."*

Necessity for caution arises also from the fact that event generators are not commercial products, developed and supported by professionals (computer scientists or programmers). Instead, the object (in this case *PYTHIA*) is "part of a one-man physics research project" (Sjöstrand 1994, 88), originally constructed to meet the author's own needs, and now available free of charge and on an "as-is" basis (i.e., in its current form) to other physicists. The author sounds a note of caution: No guarantees are given for "the proper functioning of the program, nor for the validity of physics results" (Sjöstrand 1995a, 6). Neither the epistemic (validity of physics results) nor the technological (functioning of the program) features of the code are assured. The social context of the code construction refers back to the object's character: Authors do not provide commercial software products; they are most interested in the object's epistemic features. However, they take into account that the object is needed as a tool: "Attempts are made to check processes as carefully as possible, to write programs that do not invite unnecessary errors, and to provide a detailed and accurate documentation" (ibid.). As all-round experts on the object, the authors maintain a close link between its different aspects, as is illustrated by the way the different activities are matched: the check of the physical processes modeled in the code is mentioned together with the aim to write the program in a user-friendly way. In the manuals, the different efforts directed toward the object are thus embraced and documented.

The recommendations and warnings are meant to commit the user to a particular understanding of the code: that the object's epistemic content and conceptual background define and delimit what may be called the rules of responsible usage. In the formulation of these rules, the limitations of the object need to be taken into account — which explains, for example, the recommendation that more than one event generator be used to produce results, which should then be compared.

The different cautionary tales show how the authors labor for the users' understanding and recognition of the code's epistemic dimension. Authors do not consider it necessary that users give up and exchange their object conceptions in favor of the authors' views. Instead, the authors attempt to convince the users to take their perspective on the object into account too. The tales also show how authors labor for their own recognition by presenting themselves as necessary advocates, companions, and accompanists of the objects.

## 7. Conclusions

In the preceding sections I have shown how the different object conceptions of the different actors coexist. The generators serve multiple purposes not only in the sense of being applicable to a vast array of physics scenarios (as indicated by the expression "multi-purpose generators") but also in the sense of being directed toward different goals in the research process. In different contexts the object complies with various tasks and functions, needs and desires; with its help the authors aim at reaching a deeper understanding of the model, the users aim at understanding their data and the physics behind them.[28] For the former, the object incorporates the model: by way of the object model aspects are drawn to the surface where they can be identified and studied. For the latter, the object is rather a tool to reach various epistemic goals. The mentioned physicists are all involved in epistemic activities. However, they position the object differently in the field of action with respect to the task to be accomplished. The object occupies different places in a spectrum that is spanned by the different object aspects, functions, and conceptions. The object's oscillation between the places defines a dynamics that differs from the dynamics of substitution which Rheinberger proposes, that an epistemic thing may turn into a technological object after having been intensely studied.

That no dynamics of substitution prevails is in part due to the object's innately transitory character and its capacity to unfold indefinitely. The modifications render event generators more complex: new features are added and the scope of the program is increased. Some changes "improve the program's quality." Others, such as an extension of the program's "physics range," are more tentative, due to the hypothetical character of the underlying modeling steps. Therefore, event generators remain in need of being submitted to an ongoing process of investigation and understanding. In this respect, they are "objects of knowledge" as characterized by Knorr Cetina. The object's unfolding brackets the multiplexity: Because in many cases "physics evolution has to go hand in hand with the data analysis," a new step in the object's evolution often builds on a previous step in which the object has been applied like a tool for the purpose of data analysis (i.e., physicists consider the object an epistemic thing *after* it has been employed as a technological object). To put it differently: the analysis of experimental data relies on event generators; the evolution of event generators relies on experimental data.[29] Thus event generator work exhibits a spiral dynamics. This dynamics is driven both by the discovery potential of a new experiment and by the proposition of new

[28] Both activities are not to be confounded with the endeavor to understand the code, in the language of computer scientists also known as "program understanding" or "code cognition" (see, e.g., von Mayrhauser and Vans 1995).

[29] Event generators are considered "more mature" when they have "successfully survived a number of experimental tests" (Knowles et al. 1996, 177).

modeling features. As a result the object keeps oscillating within a space that is delimited by its physics content ("epistemic object") and its black-box features ("technological object").

However, it is not only the transitory character of object versions through which a balance is maintained and may be stabilized between different object conceptions. Of importance is also the relative independence of the actors' agendas from one another. Although the activities of the various actors are linked and are to a certain extent interdependent, they also remain targeted toward different subcommunities and different research endeavors within particle physics.

The construction and ongoing evolution of the object and its successful application in experimental contexts require that all actors — authors and users alike — take the multiplex character of the object, and thus the other actors' perspectives, into consideration. In the community of particle physicists efforts are under way to further the awareness and understanding of the object's multiplex character. Cooperation and direct contact are promoted. Instances of mediation are established to minimize misunderstandings and to securely channel the flow of information. As a result, event generators remain multiplex objects, and multiplexity remains one of their constitutive elements.

## Acknowledgments

This research was funded by the Deutsche Forschungsgemeinschaft. I want to thank Deborah Dowling, Luc Gauthier, Francis Harvey, Bettina Heintz, Stefan Hirschauer, Karin Knorr Cetina, and Sergio Sismondo for valuable comments and suggestions, as well as the physicists at CERN for their help, advice, and hospitality.

## References

Bardin, Dima, and Ronald Kleiss. 1996. "Event Generators for WW Physics." In *Physics at LEP2*, vol. 2 (*CERN 96–01*), edited by Guido Altarelli, Torbjörn Sjöstrand, and Fabio Zwirner, 3–101. Geneva: CERN.

Galison, Peter L. 1995. "Contexts and Constraints." In *Scientific Practice. Theories and Stories of Doing Physics*, edited by Jed Z. Buchwald, 13–41. Chicago: University of Chicago Press.

——. 1996. "Computer Simulations and the Trading Zone." In *The Disunity of Science*, edited by P. Galison, 118–57. Stanford: Stanford University Press.

——. 1997. *Image and Logic: A Material Culture of Microphysics*. Chicago: University of Chicago Press.

James, F. 1980. "Monte Carlo Theory and Practice." *Reports on Progress in Physics* 43:1145–89.

Kling, Rob, and Walt Scacchi. 1979. "Recurrent Dilemmas of Computer Use in Complex Organizations." *AFIPS National Computer Conference* 48:107–15.

Knorr Cetina, Karin. 1997. "Sociality with Objects: Social Relations in Postsocial Knowledge Societies." *Theory, Culture and Society* 14(4):1–30.

——. 1999. *Epistemic Cultures: How the Sciences Make Knowledge.* Cambridge, Mass.: Harvard University Press.

——. Forthcoming. "Objectual Practice." In *Thinking Practices. The Practice Approach in Social Thought*, edited by K. Knorr Cetina, E. von Savigny, and T. Schatzki.

Knowles, Ian G., and Serban D. Protopopescu. 1993. "Monte Carlo Event Generators for Hadron-Hadron Collisions." In *Proceedings on the Workshop on: Physics at Current Accelerators and Supercolliders; 2–5 June 1993, Argonne National Laboratory* (ANL-HEP-CP-93-92), edited by J. L. Hewett et al.

Knowles, Ian G. et al. 1996. "QCD Event Generators." In *Physics at LEP2*, vol.2 (*CERN 96–01*), edited by Guido Altarelli, Torbjörn Sjöstrand, and Fabio Zwirner, 103–86. Geneva: CERN.

Lönnblad, Leif. 1992. "Ariadne version 4 — A Program for Simulation of QCD Cascades Implementing the Color Dipole Model." *Computer Physics Communications* 71:15–31.

Marchesini, Giuseppe et al. 1992. "HERWIG 5.1 — a Monte Carlo Event Generator for Simulating Hadron Emission Reactions with Interfering Gluons." *Computer Physics Communications* 67:465–508.

Mayrhauser, A. von, and A. M. Vans. 1995. "Program Understanding: Models and Experiments." *Advances in Computers* 40:1–38.

Merz, Martina. 1998. "'Nobody can force you when you are across the ocean' — Face to Face and e-mail Exchanges between Theoretical Physicists." In *Making Space for Science: Territorial Themes in the Shaping of Knowledge*, edited by Crosbie W. Smith and Jon Agar, 313–29. London: Macmillan.

Pickering, Andrew. 1984. *Constructing Quarks: A Sociological History of Particle Physics.* Edinburgh: Edinburgh University Press.

Rheinberger, Hans-Jörg. 1992. "Experiment, Difference, and Writing: 1. Tracing Protein Synthesis." *Studies in the History and Philosophy of Science* 23(2):305–31.

——. 1997. *Toward a History of Epistemic Things: Synthesizing Proteins in the Test Tube.* Stanford, Calif.: Stanford University Press.

Rohrlich, Fritz. 1991. "Computer Simulation in the Physical Sciences." In *PSA 1990*, vol. 2, edited by A. Fine, M. Forbes, and L. Wessels, 507–18. East Lansing, Mich.: Philosophy of Science Association.

Ruhleder, Karen. 1995. "Reconstructing Artifacts, Reconstructing Work: From Textual Edition to On-Line Databank." *Science, Technology, and Human Values* 20(1):39–64.

Sismondo, Sergio. Forthcoming. "Island Biogeography and the Multiple Domains of Models." *Biology and Philosophy.*

Sjöstrand, Torbjörn. 1992. "Monte Carlo Event Generation for LHC." In *1991 CERN School of Computing. Ystad, Sweden. 23 Augustm–2 September 1991 (CERN 92–02)*, edited by C. Verkerk.

——. 1994. "High-energy-physics event generation with PYTHIA 5.7 and JETSET 7.4." *Computer Physics Communications* 82:74–89.

——. 1995a. "PYTHIA 5.7 and JETSET 7.4. Physics and Manual." LU TP 95–20 (revised version of CERN-TH.7112/93), August.

——. 1995b. "Event Generators in Particle Physics." CERN-TH/95-10, and *Proceedings of the Fifteenth Brazilian National Meeting on Particles and Fields, Angra dos Reis, October 1994.*

Star, Susan Leigh, and James Griesemer. 1989. "Institutional Ecology, 'Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907–39." *Social Studies of Science* 19:387–420.

Traweek, Sharon. 1992. "Border Crossings: Narrative Strategies in Science Studies and among Physicists in Tsukuba Science City, Japan." In *Science as Practice and Culture*, edited by Andrew Pickering, 429–65. Chicago: University of Chicago Press.

Woolgar, Steve. 1991. "Configuring the User: The Case of Usability Trials." In *A Sociology of Monsters: Essays on Power, Technology and Domination, Sociological Review* Monograph 38, edited by John Law, 57–99. London: Routledge.

*CERN, the European Laboratory for Particle Physics, Geneva, and*
*Institute of Sociology, University of Berne*